National Aeronautics and Space Administration
Jet Propulsion Laboratory
California Institute of Technology
Pasadena, California

DD-PREF: A Language for Expressing Preferences Over Sets

Marie desJardins (University of Maryland Baltimore County) and Kiri L. Wagstaff (Jet Propulsion Laboratory, California Institute of Technology)

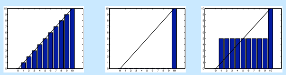
DD-PREF Language

Goal: Allow users to specify feature-based preferences over sets of objects.

For feature f, a preference statement is a tuple:

$$\mathcal{P}_f = < q_f, d_f, w_f >$$

Depth (q_f): Desired values, in terms of their *quality*.
Diversity ($d_f \in [0,1]$): Desired "spread" of values.
Weight ($w_f \in [0,1]$): Relative importance of feature f.



div = 1.00 div = 0.00 div = 0.79

Objective function (minimize):

$$\mathcal{F}_{dd}(S, \mathcal{P}, \alpha) = (1-\alpha)\,\mathrm{depth}(S,\mathcal{P}) + \alpha\,\mathrm{div}(S,\mathcal{P})$$

for subset S, preferences $\mathcal{P}$, and diversity weight α.

Algorithm: Identifying the Best Subset (Wrapper-Greedy)

Given preferences $\mathcal{P}$, a universe U of objects, a "seed" object s, and a diversity weight α, select k objects as a set.

BASIC-GREEDY($\mathcal{P}, U, s, k, \alpha$)

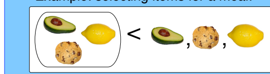
1. Initialize candidate set S with seed object $\{s\}$.
2. For j from 2 to k:
   (a) Select the object $x \in (U - S)$ that maximizes $\mathcal{F}_{dd}(S \bigcup \{x\}, \mathcal{P}, \alpha)$.
   (b) Set $S = S \bigcup \{x\}$.
3. Return S.

Wrapper-Greedy: Iterate over all possible seed objects and select the best result (by objective function value).

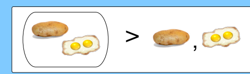
Problem Statement:

Focus: Modeling preferences over sets of items.
Why set-based methods?
Ranking items independently cannot capture inter-object interactions.

Example: selecting items for a meal:



Sub-additive utility
(redundancy, incompatibility)



Super-additive utility
(complementary)

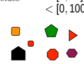
Need:
1. A language to express preferred (and non-preferred) relationships between items in a set
2. A method to select sets that satisfy the preferences
3. A method to infer preferences from user selections

Data Sets

Blocksworld: Randomly generated blocks with four features: size, color, number of sides, and location (bin number).

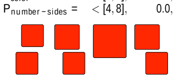
- Task 1: Create a mosaic.

	q_f	d_f	w_f
P_{size} =	< [0, 25],	0.8,	1.0 >
P_{color} =	< [0, 6],	0.75,	0.8 >
$P_{number-sides}$ =	< [3, 20],	1.0,	0.6 >
P_{bin} =	< [0, 100],	0.1,	0.6 >



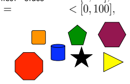
- Task 2: Build a uniform tower.

	q_f	d_f	w_f
P_{size} =	< [50, 100],	0.1,	1.0 >
P_{color} =	< [0, 6],	0.0,	1.0 >
$P_{\text{number-sides}}$ =	< [4, 8],	0.0,	1.0 >



- Task 3: Select blocks for a child.

	q_f	d_f	w_f
$\mathcal{P}_{\text{size}}$ =	< [10, 100],	1.0,	1.0 >
$\mathcal{P}_{\text{color}}$ =	< [0, 6],	1.0,	0.8 >
$\mathcal{P}_{\text{number-sides}}$ =	< [3, 20],	1.0,	0.8 >
$\mathcal{P}_{\text{bin}}$ =	< [0, 100],	0.2,	0.4 >



Mars Rover images: Collected during a field test on Earth and represented by six features: percent of the image classified as sky, rock, rock layers, light soil, dark soil, and shadow).

Users select top five of 25; the system then infers their preferences and applies them to a larger set of 100 to select their optimal top 20.

One user's top five images and the resulting derived preferences:

	q_f	d_f	w_f
$\mathcal{P}_{sky}$ =	< [33, 50],	0.94,	1.00 >
$\mathcal{P}_{rock}$ =	< [2, 16],	0.92,	1.00 >
$\mathcal{P}_{layers}$ =	< [0, 2],	0.81,	1.00 >
$\mathcal{P}_{lightsoil}$ =	< [1, 6],	0.97,	1.00 >
$\mathcal{P}_{darksoil}$ =	< [35, 47],	0.99,	1.00 >
$\mathcal{P}_{shadow}$ =	< [0, 1],	0.94,	1.00 >

Experimental Questions

1. Can we express qualitatively different preferences?
2. Given preferences and a set of items, can we efficiently select a subset to satisfy the preferences?
3. Given user selections, can we capture (learn) their implicit preferences?

Results

Experiment 1: Generate preferences for one blocksworld task, but evaluate against the goals of another task.

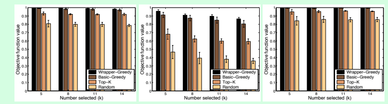
Conclusions: The best objective function values are obtained when the "right" preferences are used. Therefore, the encoded preferences *are* qualitatively different; using the "wrong" preferences impacts performance.

(Objective function values)

Generating Preference	Evaluating Preference Task 1 (mosaic)	Task 2 (tower)	Task 3 (child)
Task 1	**0.9827** $\pm$ **0.0054**	0.3054 $\pm$ 0.0306	0.9249 $\pm$ 0.0302
Task 2	0.7128 $\pm$ 0.0073	**0.8654** $\pm$ **0.0175**	0.6811 $\pm$ 0.0156
Task 3	0.8761 $\pm$ 0.0224	0.2646 $\pm$ 0.0212	**0.9994** $\pm$ **0.0007**
Random	0.7813 $\pm$ 0.0166	0.4051 $\pm$ 0.0382	0.8542 $\pm$ 0.0267

Experiment 2: Given preferences and a set of items, evaluate the ability to select a good subset. Compare Basic-Greedy, Wrapper-Greedy, Top-K (rank items independently and take the top K), and Random Selection.

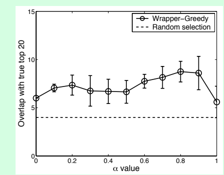
Conclusion: Wrapper-Greedy consistently finds subsets of higher value than any other method.



Mosaic Tower Child

Experiment 3: Given existing user selections, can we infer their preferences? After a user selects the top 5 of 25 images, we infer preferences based on those choices and apply them to a larger set of 100 images, to select their optimal top 20. We evaluate the quality of the retrieved subset by comparing its overlap with the images identified by the user as their true top 20 of 100. We experimented with α values from 0.0 (pure depth) to 1.0 (pure diversity).

Conclusion: The preferences enable better-than-random performance in finding the best top 20 images. The best value was obtained for $\alpha = 0.9$, suggesting that a combination of diversity and depth performs better than a sole emphasis on either one.



Averaged over 20 trials

Conclusions

1. User preferences are necessary for encoding different task goals and individual desires.
2. Feature-based preference statements can capture relevant preferences.
3. Both diversity and depth are important for finding the best subset.

Future Work:
- Investigate the use of CP-Nets to encode dependencies between features.
- Apply preferences to a large music data base, to generate DJ playlists.
- Learn preferences automatically from observing user behavior.